CS4450

Computer Networks: Architecture and Protocols

Lecture 15
Border-Gateway Protocol

Rachit Agarwal
Goal of today’s lecture

• One more level deeper dive into Border Gateway protocol
  • Recall: One of the most non-intuitive protocols in the course
    • Driven by “business goals”, rather than “performance goals”
  • Last lecture: all the intuition that I can give
    • We will recap quickly in a minute
  • Today’s lecture: some BGP details

• Important for today’s lecture:
  • We will focus on a synchronous version: one node acts at a time
    • In practice, BGP implementations are fully asynchronous

• Two goals:
  • Understand some implementation details for BGP
  • Understand why I have a love/hate relationship with BGP
Recap from last lecture
Recap: Why Peer?

Relations between ASes
- provider → customer
- peer → peer

Business Implications
- Customers pay provider
- Peers don’t pay each other

E.g., D and E talk a lot
Peering saves B and C money
Recap: Inter-domain Routing Follows the Money

- ASes provide “transit” between their customers
- Peers do not provide transit between other peers
- An AS does not provide transmit between provider and peers

Traffic allowed: A → B → C
Traffic not allowed: Q → R → A, D → A → B → E → F, E → D
Recap: Inter-domain Routing Setup

- Destinations are IP prefixes (12.0.0.0/8)

- Nodes are Autonomous Systems (ASes)
  - Internals of each AS are hidden

- Links represent both physical links and business relationships

- BGP (Border Gateway Protocol) is the Interdomain routing protocol
  - Implemented by AS border routers
Recap: BGP is Inspired by Distance Vector

- Per-destination route advertisements
- No global sharing of network topology
- Iterative and distributed convergence on paths
- But, four key differences
  - BGP does not pick shortest paths
  - Each node announces a PATH per destination
  - Selective Route advertisement: not all paths are announced
  - BGP may aggregate paths
    - may announce one path for multiple destinations
Questions?
A modification of the solution for the stable paths problem called \textsc{good gadget} is illustrated in Fig. 3(a). The specification has no solution and the SPVP protocol will always diverge. However, as is explained in Section IV, the protocol can diverge for this problem. Finally, by reordering the ranking of paths at node 4, we produce a specification called \textsc{naughty gadget} which is shown in Fig. 3(c). No other path assignments are stable for this problem.
A stable path assignment is a specification of paths to reach each node, with the highest ranked path at the bottom going down to the lowest ranked nonempty path at the vertical list next to the node. The ranking function for each nonzero node is depicted as a stable path assignment in the figure. A stable path assignment implicitly defines a tree rooted at the origin. Note, however, that this is not always a spanning tree. For example, the solution for Fig. 2(a) presents a stable paths tree that is illustrated in Fig. 1(a). In both cases, the ranking functions prefer shorter paths to longer paths and the solutions are shortest path trees. If no such assignment exists, then the solution for Fig. 2(b) is not a shortest path tree. This results in one shortest path tree as shown in Fig. 2(c). The solution for Fig. 2(d) presented in Fig. 3(b) is an alternative solution, called BAD GADGET (3 4 2 0) for node 3, yet it has the same unique solution as GOOD GADGET (3 4 2 0) for node 3, as illustrated in Fig. 3(c). No other path assignments are stable for this problem.

So far, our examples each has had at most one solution. This is illustrated in the figure, which shows that having more than one solution is illustrated in Fig. 3(a). The stable path assignment having more than one solution is illustrated in Fig. 3(a). The ranking at node 4 breaks ties between paths of equal length. This results in one shortest path tree as illustrated in Fig. 3(b). An alternative solution is illustrated in Fig. 3(c). No other path assignments are stable for this problem. Therefore, any stable path assignment is solvable if there is at least one solution.

### Table 1: BGP Example (Naughty good)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>-</td>
</tr>
<tr>
<td>R2</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>430</td>
</tr>
<tr>
<td>R3</td>
<td>130</td>
<td>20</td>
<td>30</td>
<td>430</td>
</tr>
</tbody>
</table>

### NAUGHTY GADGET

- Node 1
- Node 2
- Node 3
- Node 4

### Table 1: BGP Example (Naughty good)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>-</td>
</tr>
<tr>
<td>R2</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>430</td>
</tr>
<tr>
<td>R3</td>
<td>130</td>
<td>20</td>
<td>30</td>
<td>430</td>
</tr>
</tbody>
</table>
BGP Details

- BGP Policy
  - Typical policies and implementation

- BGP protocol details

- Why do I love BGP?

- Why am I not blindly in love with BGP?
**Policy:**

*Imposed in how routes are selected and exported*

- **Selection**: Which path to use
  - Controls whether / how traffic *leaves* the network
- **Export**: Which path to advertise
  - Controls whether / how traffic *enters* the network
Typical Selection Policy

- In decreasing order of priority:
  1. Make or save **money** (send to customer > peer > provider)
  2. Maximize **performance** (smallest AS path length)
  3. Minimize use of my **network bandwidth** ("hot potato")
  4. ...
## Typical Export Policy

<table>
<thead>
<tr>
<th>Destination prefix advertised by...</th>
<th>Export route to...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer</td>
<td>Everyone (providers, peers, other customers)</td>
</tr>
<tr>
<td>Peer</td>
<td>Customers</td>
</tr>
<tr>
<td>Provider</td>
<td>Customers</td>
</tr>
</tbody>
</table>

Known as the “Gao-Rexford” rules
Capture common *(but not required!)* practice
With Gao-Rexford, the AS policy graph is a DAG (directed acyclic graph) and routes are “valley free”
BGP Details

- BGP Policy
  - Typical policies and implementation

- BGP protocol details

- Why am I not blindly in love with BGP?

- Why do I love BGP?
Who speaks BGP?

Border routers at an Autonomous System
What Does “speak BGP” Mean?

- Implement the BGP Protocol Standard
  - Internet Engineering Task Force (IETF) RFC 4271

- Specifies what messages to exchange with other BGP “speakers”
  - Message types (e.g. route advertisements, updates, withdrawals)
  - Message syntax

- Specifies how to process these messages
  - When you receive a BGP update, do x
  - Follows BGP state machine in the protocol spec and policy decisions, etc.
A border router speaks BGP with border routers in other ASes
A border router speaks BGP with other (interior and border) routers in its own AS
**eBGP, iBGP, IGP**

- **eBGP**: BGP sessions between border routers in **different** ASes
  - Learn routes to external destinations

- **iBGP**: BGP sessions between border routers and other routers within the **same** AS
  - Distribute externally learned routes internally

- **IGP**: Interior Gateway Protocol = **Intradomain routing protocol**
  - Provides internal reachability
  - e.g. link-state, distance-vector, etc.
1. Provide internal reachability (IGP)
2. Learn routes to external destinations (eBGP)
3. Distribute externally learned routes internally (iBGP)
4. Travel shortest path to egress (IGP)
Basic Messages in BGP

- **Open**
  - Establishes BGP session

- **Update**
  - Inform neighbor of **new routes** *(announcements)*
  - Inform neighbor of **old routes** that become inactive *(withdrawals)*

- **Keepalive**
  - Inform neighbor that connection is still viable
Route Updates

- Format: `<IP prefix: route attributes>`

- Two kinds of updates:
  - **Announcements**: new routes or changes to existing routes
  - **Withdrawals**: remove routes that no longer exist

- Route Attributes
  - Describe routes, used in selection/export decisions
  - Some attributes are **local**
    - i.e. private within an AS, not included in announcements
  - Some attributes are **propagated** with eBGP route announcements
  - Many standardized attributes in BGP
Route Attributes (1): ASPATH

- Carried in route announcements
- Vector that lists all the ASes a route advertisement has traversed (in reverse order)
Route Attributes (2): LOCAL PREF

- “Local Preference”
- Used to choose between different AS paths
- The higher the value, the more preferred
- Local to an AS; carried only in iBGP messages

BGP table at AS4:

<table>
<thead>
<tr>
<th>Destination</th>
<th>AS Path</th>
<th>Local Pref</th>
</tr>
</thead>
<tbody>
<tr>
<td>140.20.1.0/24</td>
<td>AS3 AS1</td>
<td>300</td>
</tr>
<tr>
<td>140.20.1.0/24</td>
<td>AS2 AS1</td>
<td>100</td>
</tr>
</tbody>
</table>
Route Attributes (3) : MED

- “Multi-Exit Discriminator”
- Used when ASes are interconnected via two or more links
- Specifies how close a prefix is to the link it is announced on
- Lower is better
- AS announcing prefix sets MED
- AS receiving prefix (optionally!) uses MED to select link
Route Attributes (4): IGP Cost

- Used for **hot-potato routing**
- Each router selects the closest egress point based on the path cost in intra-domain protocol
Using Attributes

- Rules for route selection in priority order

  1. Make or save money (send to customer > peer > provider)
  2. Maximize performance (smallest AS path length)
  3. Minimize use of my network bandwidth ("hot potato")
  4. ...
## Using Attributes

- Rules for route selection in priority order

<table>
<thead>
<tr>
<th>Priority</th>
<th>Rule</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LOCAL PREF</td>
<td>Pick highest LOCAL PREF</td>
</tr>
<tr>
<td>2</td>
<td>ASPATH</td>
<td>Pick shortest ASPATH length</td>
</tr>
<tr>
<td>3</td>
<td>MED</td>
<td>Lowest MED preferred</td>
</tr>
<tr>
<td>4</td>
<td>eBGP &gt; iBGP</td>
<td>Did AS learn route via eBGP (preferred) or iBGP?</td>
</tr>
<tr>
<td>5</td>
<td>iBGP path</td>
<td>Lowest IGP cost to next hop (egress router)</td>
</tr>
<tr>
<td>6</td>
<td>Router ID</td>
<td>Smallest next-hop router’s IP address as tie-breaker</td>
</tr>
</tbody>
</table>
BGP Details

- BGP Policy
  - Typical policies and implementation

- BGP protocol details

- Why am I not blindly in love with BGP?

- Why do I love BGP?
Why am I not blindly in love with BGP?

- Reachability
- Convergence
- Performance
- Anomalies
- Security
Reachability

- In shortest-path routing, if graph is connected then reachability is assured.
- With policy routing, this doesn’t always hold.
Convergence

- If all AS policies follow Gao-Rexford rules,
  - Then BGP is guaranteed to converge (safety)

- For arbitrary policies, BGP may fail to converge!

- Do you appreciate the beauty of the result (and Gao-Rexford policies?)
Example of Policy Oscillation

“1” prefers “1 3 0” over “1 0” to reach “0”
Initially: nodes 1, 2, 3 know only shortest path to 0
1 advertises its path 1 0 to 2
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

3 advertises its path 3 0 to 1
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

1 withdraws its path 1 0 from 2
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

2 advertises its path 2 0 to 3

advertise: 2 0
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

3 withdraws its path 3 0 from 1
Step-by-step Policy Oscillation
1 advertises its path 1 0 to 2
Step-by-step Policy Oscillation
Step-by-step Policy Oscillation

2 withdraws its path 2 0 from 3

withdraw: 2 0
Step-by-step Policy Oscillation

We are back to where we started!
A modification of the solution for the shortest paths problem called \textit{STABLE PATHS PROBLEM AND INTERDOMAIN ROUTING} is shown in Fig. 2(c). If no such assignment exists, then the solution for the shortest paths problem is unsolvable. A stable path assignment is illustrated in Fig. 1(b). If we reverse the ranking order of paths at node 4, we produce a specification called \textit{GOOD GADGET} and its two solutions. However, as is explained in Section IV, the protocol \textit{SPVP} can diverge for this problem. Finally, by reordering the ranking of paths at node 4, we produce a specification called \textit{NAUGHTY GADGET} and its two solutions.

So far, our examples each has had at most one solution. This is not always the case. The simplest instance, called \textit{BAD GADGET}, is illustrated in Fig. 3(a). The solution and the \textit{SPVP} protocol will always diverge. If there is a stable path assignment having more than one solution, then the solution is not unique. No other path assignments are stable for the shortest path tree. This results in one shortest path tree as illustrated in Fig. 2(b) is not a shortest path tree. This is the case if there is a node with two or more paths of equal length. This results in one shortest path tree as illustrated in Fig. 3(c). No other path assignments are stable for the shortest path tree.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
 & 1 & 2 & 3 & 4 \\
\hline
R1 & 10 & 20 & 30 & - \\
\hline
R2 & 10 & 20 & 30 & 420 \\
\hline
R3 & 10 & 20 & 3420 & 420 \\
\hline
R4 & 10 & 210 & 3420 & 420 \\
\hline
R5 & 10 & 210 & 3420 & - \\
\hline
R6 & 10 & 210 & 30 & - \\
\hline
R7 & 130 & 210 & 30 & - \\
\hline
R8 & 130 & 20 & 30 & - \\
\hline
R9 & 130 & 20 & 30 & 420 \\
\hline
R10 & 130 & 20 & 3420 & 420 \\
\hline
R11 & 10 & 20 & 3420 & 420 \\
\hline
\end{tabular}
\end{table}
The ranking of paths is not required to prefer shorter paths to longer paths. For example, Fig. 2(a) presents a stable paths problem called the "Naughty Gadget." A modification of this problem results in one shortest path tree, as illustrated in Fig. 2(b). An alternative solution for this problem is shown in Fig. 2(c). No other path assignments are stable for this problem. Finally, by reordering the nodes, one can obtain another stable assignment, presented in Fig. 2(d). This specification has no unique solution to this problem.
Performance Non-Issues

- Internal Routing
  - Domains typically use “hot potato” routing
  - Not always optimal, but economically expedient

- Policy not about performance
  - So policy-chosen paths aren’t shortest

- AS path length can be misleading
  - 20% of paths inflated by at least 5 router hops
• AS path length can be misleading
  • An AS may have many router-level hops

Performance (example)

BGP says that path 4 1 is better than path 3 2 1
Performance: Real Issue

Slow Convergence

- BGP outages are biggest source of Internet problems

- Labovitz et al. *SIGCOMM’97*
  - 10% of routes available less than 95% of the time
  - Less than 35% of routes available 99.99% of the time

- Labovitz et al. *SIGCOMM 2000*
  - 40% of path outages take 30+ minutes to repair

- But most popular paths are very stable
Security

- An AS can claim to serve a prefix that they actually don’t have a route to (blackholing traffic)
  - Problem not specific to policy or path vector
  - Important because of AS autonomy
  - *Fixable*: make ASes prove they have a path

- But...

- AS may forward packets along a route different from what is advertised
  - Tell customers about a fictitious short path...
  - Much harder to fix!
BGP Outline

- BGP Policy
  - Typical policies and implementation

- BGP protocol details

- Why am I not blindly in love with BGP?

- Why do I love BGP?
Think about what BGP is able to achieve....

• Every ISP acts completely independently and selfishly....
  • Policies, path selection, path advertisement, complete privacy
  • Creates relationships independently
  • Cares only about its own interest—money earned
  • Operates in a distributed manner
  • And yet ....

• How frequently are we disconnected from the Internet?
  • Almost rare ....
  • It just works .... And ...
BGP is one of those examples where ....

- Real-world protocol that was born in an organic manner ...
  - ... has led to advances in computing
  - A recent result:
    - BGP is an example of the “hardest” Nash equilibrium instance
    - Many recent beautiful results in theoretical computer science
  - But ...

- My real love?
  - It just works .... And nobody knows why!!

- Has been working since 1991(5)
  - With almost no changes ...
  - Perfect in inception, but perhaps poor in execution
Now you know as much as my PhD students :-)
BGP Update Processing

Open ended programming. Constrained only by vendor configuration language

Control plane

Data plane

IP Forwarding Table

Forwarding Entries

Data packets

BGP Updates

Data packets